MATLAB commands

Here's some helpful commands to get started, MATLAB has great help features,
>>help command

and it will decribe the usage of the command and give you the various flags, and a list of other related commands i.e.
>>help plot
MATLAB is very powerful and can do many complicated functions, just browse help.

DATA Input/Output

- **load**
  - **Check your directory, then >load mydata . As long as it is in numeric form, some n x m matrix of numbers, it will save the data into the variable mydata**
- **save**
  - **>save myvars This will save the data into a .mat file, which is really only useful for reading back into MATLAB**
  - **>save namefile myvars –ASCII This will save the values in myvars, array, matrix, whichever, into the file namefile, and this can then be accessed as a normal number file in emacs for instance.**

To display more digits than just the default 4, use

>>format long

Most UNIX commands also work in MATLAB
>>ls

MATRICES/ARRAYS

To make data to play with easily, generate random numbers

>>A=rand(rows,columns)
This will give you the matrix A with your choice of rows and columns

>> A=rand(5,4)

A =

0.9501	**0.7621**	**0.6154**	**0.4057**
0.2311	**0.4565**	**0.7919**	**0.9355**
0.6068	**0.0185**	**0.9218**	**0.9169**
0.4860	**0.8214**	**0.7382**	**0.4103**
0.8913	**0.4447**	**0.1763**	**0.8936**

To access certain parts of the array, say rows 1,2,3 of column 1 use:

```
>> A(1:3,1)

ans =

    0.9501
    0.2311
    0.6068
```

or say row 2, all columns use

```
>> b=A(2,:)

b=

    0.2311    0.4565    0.7919    0.9355
```

So the colon is shorthand for all columns. Works the same for all rows.

TRANSPOSE:
Use the apostrophe

```
>> b'

ans =

    0.2311
    0.4565
    0.7919
    0.9355
```

MULTIPLICATION

Default is matrix multiplication, so to multiply A*A, the inner dimensions must agree, an N x M matrix multiplies a M x L matrix

In order to multply each element in the array , e.i. you want A^2,
You must use a period, so it does not try to do matrix multiply
b^2 would be

```
>>b.*b

ans =

    0.0534    0.2084    0.6272    0.8751
```

or

>>b.^2
ans =

0.0534 0.2084 0.6272 0.8751

These same rules hold for division.
FUNCTIONS
To create functions, you must declare the numerical arrays of the independent variable(s)
>>x=0:0.01:1
the syntax is x=start:deltax:end

(to suppress the output, put a semicolon at the end.)

example functions:
>>y=sin(x)
>>y=x.^3+x.^2+x

PLOTTING
For 2D plots
>>plot(x, y, 'r*-')
will plot the curve, with a red star at each point and a line connecting
you can change the color, (r, k, b, c, y, g) and the points (*, +, etc). Remove the line to see just points. Use help command for more choices.

To keep the data on will plotting more curves use
>>figure(1)
>>hold on
>>plot(x, z, 'b*-')
This curve will then be added on
>>hold off
stops that command

You can generate subplots of one figure using
>>subplot(3,1,1)
which will make a plot with 3 rows and 1 column of subplots
you access the different plots with the 3rd indice, the second suplot would be
>>subplot(3,1,2)
You can change axis limits
>>axis([xstart xend ystart yend])
and add labels
>>xlabel(' x axis')
>>ylabel(' my y axis label')
>>title(' Sin(x) plot')

For 3D plots
>>plot3(x,y,z)
or If you have a N x M matrix you want to visualize, surf or mesh will work
>>surf(N_array, M_array, NMmatrix)

LOOPS/LOGIC

To generate for loops, MATLAB indexing starts at 1!

```
for i=1:20
        x(i+1)=x(i)*2;
end
```

To use if commands

```
if(b==0)
        b=b+1
else
        b=b*4
end
```

not if is denoted by: if(b~=0)

MATLAB PROGRAMS

Open up a blank matlab document, and when you save it: myprog.m
That will be the name you use on the command line. You basically make functions

Inside the .m file:

```
function[a, b, c]=myprog(num)

%this is how you make comments, I'm just going to make a random matrix of
%dimensions num x num

A=rand(num, num);

% You don't need the semicolon like in c++, putting it there just prevents
%MATLAB from printing this out.

a=rand(1,:);
b=rand(2,:);
c=rand(3,:);

%that's all you need for a program! To run it:

>>[x,y,z]=myprog(4)
```

x =

0.0579 0.3529 0.8132 0.0099

y =

0.1389 0.2028 0.1987 0.6038

z =

0.2722 0.1988 0.0153 0.7468

>>

And those variables will be saved into your workspace.

SYMBOLIC TOOLBOX

If you have the Symbolic toolbox, MATLAB can do integration, differentiation for you.

```
>>syms x
>>y=x^2
>>int(y)
ans =

1/3*x^3
```